Hit List

Clear | Generate Collection | Print | Fwd Refs | Bkwd Refs | Generate OACS

Search Results - Record(s) 1 through 8 of 8 returned.

☐ 1. Document ID: US 6138216 A

Using default format because multiple data bases are involved.

L11: Entry 1 of 8 File: USPT Oct 24, 2000

US-PAT-NO: 6138216
DOCUMENT-IDENTIFIER: US 6138216 A

TITLE: Microprocessor cache consistency

DATE-ISSUED: October 24, 2000

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Harvey; Ian Nigel	Cambridge			GB

US-CL-CURRENT: 711/139; 711/138, 711/144

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Draw De

☐ 2. Document ID: US 5933845 A

L11: Entry 2 of 8 File: USPT Aug 3, 1999

DOCUMENT-IDENTIFIER: US 5933845 A
**** See image for Certificate of Correction ****
TITLE: Flash memory management method including shifting or copying data to other blocks to optimize storage space and allow entire blocks to be erased

Brief Summary Text (8):
Another task of the invention, in addition to the extensive loading, is to provide a flash memory management that is user friendly. This is achieved by a method for operating a flash memory, which is divided into blocks, and where the blocks are subdivided into a plurality of segments, wherein data are stored in the form of files in segments and files are marked for erasure in segments, the method comprising the steps of determining an up-to-date degree of filling for the flash memory, wherein the number of occupied segments marked for erasure, and the number of unmarked and unoccupied segments are determined for each block, and performing a memory optimization after a degree of filling threshold has been exceeded.

Brief Summary Text (9):
It is also achieved by a memory encoded with executable instructions representing a

h e b b g e e e f e f g ef b e

program module for the operation of a flash memory divided into blocks, and whose blocks are subdivided into a number of segments, by which a computer is instructed to determine an allowable degree of flash memory filling by the number of unoccupied, occupied and marked for erasure segments, and through occupied and unmarked segments of each block, and a memory optimization is then performed by means of control commands, the control commands comprising: (1) a first such control command causes segments occupied by files to be shifted, and (2) a second such control command erases remaining occupied segments which are marked for erasure, or (3) a third such control command wherein a block is determined in which no occupied and unmarked segments are located, which is then erased, or wherein a fourth control command causes determination of the block containing the smallest number of occupied unmarked segments and copying of the occupied unmarked segments to unoccupied segments of other blocks, or wherein a fifth control command causes determination of the block containing the smallest number of occupied unmarked segments and temporary storage of the occupied unmarked segments in marked segments of other blocks, and causes the erasure of the determined block, and causes reconstruction of the occupied unmarked segments or wherein a sixth control command causes a user inquiry, in which the user can mark files for erasure.

Brief Summary Text (10):
A further task of the invention is achieved by the use of such a memory encoded with executable instructions representing a program module for optimizing the storage of voice messages, particularly in a telephone answering machine. A still further task of the invention is achieved by a computer-readable storage medium having a program or structured data recorded thereon for the operation of a flash memory divided into blocks, and whose blocks are subdivided into a plurality of segments, by which a computer is instructed to determine an allowable degree of flash memory filling through the number of unoccupied, occupied and marked for erasure segments, and through occupied and unmarked segments of each block, and a memory optimization is then performed by means of one or more of the following control commands, wherein a control command causes segments occupied by files to be shifted, and wherein the remaining occupied marked for erasure segments are erased by means of a control command, or wherein a block is determined in which no occupied and unmarked segments are located, which is then erased by means of a control command, or wherein a block containing the smallest number of occupied unmarked segments is determined, which are then copied to unoccupied segments of other blocks by means of a control command, or wherein the block containing the smallest number of occupied unmarked segments is determined, which are then temporarily stored in marked segments of other blocks by means of a control command, and are then reconstructed after the determined block has been erased, or wherein the block containing the smallest number of occupied unmarked segments is determined, which are then copied to marked-for-erasure segments of another block using a logic operation to combine data of the unmarked segments with data of the marked segments, and/or wherein another control command causes a user inquiry, in which the user can mark files for erasure.

Detailed Description Text (11):
Furthermore, at the start of a file, another area III for example contains the total length of the existing file to its end. As an option, the date, year and time of the file entry can also be recorded. After that, the data to be saved is stored in the remaining area or areas. This clearly establishes which segments are occupied. To mark a file for erasure, a zero is entered e.g. into all segment initiation labels II, thus 00000. This clearly establishes that the data stored therein are ready for erasure. A code in the segment initiation label II can establish which type of data are stored therein; for example pulse code modulation (PCM) data or adaptive differential pulse code modulation (ADPCM) data etc., which is indicated by a code 01000 or 00100.

Detailed Description Text (12):
In the following, a memory encoded with executable instructions representing a

program module, herein referred to as "program module", is explained (without drawing) for the operation of a flash memory. The flash memory is also divided into blocks, which in turn are subdivided into a number of segments. An allowable degree of flash memory filling (occupancy level) is also determined, resulting from the number of unoccupied segments, the number of occupied segments marked for erasure, and the number of occupied unmarked segments of each block. A memory optimization then takes place by means of subsequent control commands. One control command causes segments containing files to be shifted, and another control command causes the erasure of a block with the remaining occupied segments marked for erasure. In the event that this cannot be accomplished, a block in which no occupied unmarked segments are located is determined, and the entire block is then erased by means of a control command. If this is not possible either, the block containing the smallest number of occupied unmarked segments is determined.

CLAIMS:

1. A method for operating a flash memory, which is divided into blocks, and where the blocks are subdivided into a plurality of segemnts, wherein

data are stored in the form of files in segments, and files are marked for erasure in segments, said method comprising the steps of

for each file marked for erasure, storing information in each segment which contains the data of the file. the information identifying the segment for erasure ,

determining an up-to-date degree of filling for the flash memory, wherein the number of occupied segments marked for erasure, and the number of unmarked and unoccupied segments are determined for each block, and

performing a memory optimization after a degree of filling threshold has been exceeded.

11. A memory encoded with executable instructions representing a program module for the operation of a flash memory divided into blocks, and in which blocks are subdivided into a number of segments, the segments comprising information identifying whether a particular segment is to be erased, the executable instructions representing the program module being capable of instructing a computer to determine an allowable degree of flash memory filling by the number of unoccupied, occupied and marked for erasure segments, and by occupied and unmarked segments of each block, and being capable of instructing the computer to perform a memory optimization by means of control commands,

wherein a first control command causes segments occupied by files to be shifted, and a second control command erases remaining occupied segments which are marked for erasure, and in the event that files cannot be shifted,

a third control command causes determination of a block in which no occupied and unmarked segments are located, which is then erased, and in the event that no block can be determined in which no occupied and unmarked segments are located,

a fourth control command causes determination of the block containing the smallest number of occupied unmarked segments and copying of the occupied unmarked segments to unoccupied segments of other blocks, and in the event that occupied unmarked segments cannot be copied to unoccupied segments of other blocks

a fifth control command causes determination of the block containing the smallest number of occupied unmarked segments and temporary storage of the occupied unmarked segments in marked segments of other blocks, and causes the erasure of the determined block, and causes reconstruction of the occupied unmarked segments and

h e b b g e e e f e f g ef b e

in the event that the block containing the smallest number of occupied unmarked segments cannot be temporarily stored in marked segments of other blocks a sixth control command causes a user inquiry, in which the user can mark files for erasure.

12. In a telephone answering machine, a memory encoded with executable instructions representing a program module for the operation of a flash memory divided into blocks, and in which blocks are subdivided into a number of segments, the segments comprising information identifying whether a particular segment is to be erased, the executable instructions representing the program module being capable of instructing a computer to determine an allowable degree of flash memory filling by the number of unoccupied, occupied and marked for erasure segments, and by occupied and unmarked segments of each block, and being capable of instructing the computer to perform a memory optimization by means of control commands,

wherein a first control command causes segments occupied by files to be shifted, and a second control command erases remaining occupied segments which are marked for erasure, and in the event that files cannot be shifted,

a third control command causes determination of a block in which no occupied and unmarked segments are located, which is then erased, and in the event that no block can be determined in which no occupied and unmarked segments are located,

a fourth control command causes determination of the block containing the smallest number of occupied unmarked segments and copying of the occupied unmarked segments to unoccupied segments of other blocks, and in the event that occupied unmarked segments cannot be copied to unoccupied segments of other blocks

a fifth control command causes determination of the block containing the smallest number of occupied unmarked segments and temporary storage of the occupied unmarked segments in marked segments of other blocks, and causes the erasure of the determined block, and causes reconstruction of the occupied unmarked segments and in the event that the block containing the smallest number of occupied unmarked segments cannot be temporarily stored in marked segments of other blocks a sixth control command causes a user inquiry, in which the user can mark files for erasure.

13. A method for optimizing storage space in a flash memory divided into blocks, and where the blocks are subdivided into a plurality of segments, the segments comprising information identifying whether a particular segment is to be erased, wherein

after a degree of filling threshold has been exceeded, as determined by the number of occupied segments marked for erasure, and the number of occupied unmarked segments and unoccupied segments of each block, determining the block containing the smallest number of occupied unmarked segments,

shifting occupied unmarked segments of this block to segments of another block, which are marked for erasure, using a logic operation to combine the data of the occupied unmarked segments with the data of the segments marked for erasure,

erasing the previously determined block containing the smallest number of occupied unmarked segments.

19. A memory encoded with executable instructions representing a program module for the operation of a flash memory divided into blocks, and in which blocks are subdivided into a plurality of segments, the segments comprising information identifying whether a particular segment is to be erased, the executable instructions representing the program module being capable of instructing a computer to determine an allowable degree of flash memory filling through the

number of unoccupied, occupied and marked for erasure segments, and by occupied and unmarked segments of each block, and being capable of instructing the computer to perform a memory optimization by means of control commands,

wherein a first control command causes segments occupied by files to be shifted, and wherein the remaining occupied marked for erasure segments are erased by means of a second control command, and in the event that files cannot be shifted,

a block is determined in which no occupied and unmarked segments are located, which is then erased by means of a third control command, and in the event that no block can be determined in which no occupied and unmarked segments are located,

a block containing the smallest number of occupied unmarked segments is determined, which are then copied to unoccupied segments of other blocks by means of a fourth control command, and in the event that occupied unmarked segments cannot be copied to unoccupied segments of other blocks, the block containing the smallest number of occupied unmarked segments is determined, which are then temporarily stored in marked segments of other blocks by means of a fifth control command, and are then reconstructed after the determined block has been erased, and in the event that occupied unmarked segments cannot be copied to unoccupied segments of other blocks,

the block containing the smallest number of occupied unmarked segments is determined, which are then copied to marked-for-erasure segments of another block, using a logic operation to combine data of the unmarked segments with data of the marked segments by means of a sixth control command, and in the event that the block containing the smallest number of occupied unmarked segments cannot be temporarily stored in marked segments of other blocks,

a seventh control command causes a user inquiry, in which the user can mark files for erasure.

☐ 3. Document ID: US 5564033 A

L11: Entry 3 of 8 File: USPT Oct 8, 1996

DOCUMENT-IDENTIFIER: US 5564033 A
TITLE: Disk array system using detachable memory medium

Detailed Description Text (12):
Because of these possibilities, it is clear that the read and write of data according to a predetermined access sequence established for use with a fixed external memory unit is not applicable to that of a memory device having detachable memory media. Therefore, the invention addresses the above-identified possibilities and takes measures to prevent these possibilities from occurring. Such measures will be described in the embodiments set forth according to the following.

CLAIMS:

1. A peripheral memory unit for use with a data processing system, comprising:

said memory unit having a plurality of detachable memory media for storage of data;

h e b b g e e e f e f g ef b e

means for dividing units of data to be stored into a series of divided subunits of data;

means for initializing each of said memory media including storing reference information in each of said memory media to be used for restoring said divided subunits of data into said units of data during a read operation;

means for writing in parallel said subunits of data in accordance with said reference information among said plurality of memory media;

means for reading said reference information stored in said memory media;

means for reading in parallel said divided subunits of data in accordance with said reference information read from said plurality of memory media; and

means for receiving and restoring said divided subunits of data read by said divided data reading means and for outputting said divided subunits as said units of data;

wherein said writing means writes said subunit of data in a sequence and;

wherein said means for reading said reference information in response to receiving a read/write instruction first reads said group information from each of said memory media to determine if all said memory media in said memory unit are part of said predetermined group of said memory media required for executing the read/write instruction, and then reads said partial sequence data when it is determined that all of said memory media required for executing the read/write instruction are present; wherein said means for reading said reference information includes means for determining an access sequence from said partial sequence data; and further wherein said means for reading said subdivided data reads said subunits from said memory media in accordance with said access sequence.

4. A memory unit according to claim 2, wherein said writing means writes said divided subunits of data to said predetermined group of memory media in a sequence and said initializing means writes partial sequence data in each of said memory media identifying a place in said sequence that each said memory media has been assigned by said writing means as said reference information.

7. A peripheral memory unit for use with a data processing system, comprising:

said memory unit having a central processing unit and a plurality of detachable memory media for storage of data;

means for dividing units of data to be stored into a series of divided subunits of data;

means for initializing each of said memory media including generating and storing reference information including group information identifying a predetermined group of said plurality of memory media to which data is to be written to or read from each of said memory media to be used for reading and restoring said divided subunits of data;

means for writing in parallel said subunits of data in a sequence among said group of memory media;

means for reading said reference information and for reading in parallel said divided subunits of data from said group of memory media in accordance with said reference information and for restoring said divided subunits into said units of data;

wherein said means for reading said reference information, in response to receiving a read or write instruction, determines if all of said memory media in said memory unit are part of said predetermined group of said memory media required for executing the read or write instruction; and

wherein said central processing unit suspends execution of said read or write instruction when one of said memory media is determined to not be part of said predetermined group;

wherein said means for reading said reference information reads said partial sequence data and determines an access sequence from said partial sequence data; and further wherein said means for reading said subunits of divided data reads said subunits from said memory media in accordance with said access sequence.

17. A peripheral memory unit for use with a data processing system, comprising:

said memory unit having a plurality of means for receiving respective detachable memory media for storage of data;

means for dividing units of data to be stored into a series of divided subunits of data;

means for initializing each of said memory media including generating and storing reference information including group information identifying a predetermined group of said plurality of memory media to which data is to be written to or read from reference information in each of said memory media to be used for restoring said divided subunits of data into said units of data during a read operation;

means for writing in parallel said subunits of data in a sequence to each of said memory media received in said receiving;

means, in response to receiving a read or write instruction, for reading said reference information of each of said respective memory media received in said plurality of receiving means to determine if all of said memory media in said predetermined group are received in said corresponding receiving means; and

said central processing unit suspending execution of said read/write instruction when at least one of said memory media of said predetermined-group is determined to not be in a corresponding one of said receiving means;

wherein said means for reading said reference information, in response to receiving a read or write instruction, reads said group information from each of said memory media to determine if all of said memory media in said memory unit are part of said predetermined group of said memory media required for executing the read/write instruction, and also reads said partial sequence data for determining an access sequence used by said reading means for reading said subunits of data in said sequence in which said subunits were written by said writing means.

☐ 4. Document ID: US 5321828 A

L11: Entry 4 of 8 File: USPT Jun 14, 1994

h e b b g e e e f e f g ef b e

DOCUMENT-IDENTIFIER: US 5321828 A
TITLE: High speed microcomputer in-circuit emulator

Detailed Description Text (88):
The trace memory 62 can be divided into the following subsections:

Detailed Description Text (190):
Software breakpoints are implemented by inserting an fmark instruction in the source code at the desired breakpoint location. When clearing, the breakpoint location is replaced with original data. There are four different types of hardware breakpoints: address, data, range, and special. ICE 10 provides two address breakpoints, two data breakpoints, one range breakpoint, and four 10 special breakpoints that can be set to break on branch, call, return, and supervisor instructions. Another control command, TP, allows the trace position count to be set in the range of zero to 65,535, thereby allowing execution to continue "n" number of cycles after a break condition has been recognized.

Detailed Description Text (760):
DATA clears one or both data breakpoints internal to the Intel i960 CA chip.

Detailed Description Text (773):
DATA clears the range breakpoint on the target system 14 data bus.

Detailed Description Text (961):
BUS 24 trace buffer 202 is controlled by the multi-level trigger and store control mechanism 208. Besides serving as input for trace reconstruction, BUS 24 trace data can also be used for performance analysis such as FUNCTION level code coverage and histograms. BUS 24 trace memory 202 can be divided into the following subsections:

Detailed Description Text (1070):
The time-tag trace memory can be divided into the following subsections:

Detailed Description Text (1199):
A WRITE to these addresses (xD000FH and xD00014H) with dummy data causes the trace Counter address to be CLEARED. The CP issues this command when it wants to capture target system 14 data into the trace buffer 202 starting at location 0. A WRITE to this address (xD0015H) with dummy data causes BUS 24 trace Counter address (on-board 2) to be CLEARED. The CP issues this command when it wants to capture BUS 24 data into BUS 24 trace buffer 202 starting at location 0. The HWLOCK device allows ICE 100 software to READ ICE 100 hardware ID for the purpose of locking a copy of software to a particular ICE 100 hardware. The control processor (PC/AT 20) can read up to 7 bytes of hard coded ID from the HWLOCK device based on the following table.

☐ 5. Document ID: US 4769636 A

L11: Entry 5 of 8 File: USPT Sep 6, 1988

DOCUMENT-IDENTIFIER: US 4769636 A
TITLE: Display control method for multi-window system

h e b b g e e e f e f g ef b e

CLAIMS:

1. A method of display control for a multi-window system provided with a display screen in which a plurality of rectangular windows are set, first memory means for storing display data in positions corresponding to the windows on the display screen, second memory means for storing a plurality of display data of a virtual screen corresponding to each of said windows which displays data included in the partial region on the virtual screen corresponding thereto, third memory means for storing definition data representing the positional data on the display screen for each window and the correspondence thereof with the virtual screen, and control means for partially rewriting the display data stored in said first memory means;

the method comprising:

a first step of designating one of the windows in the display screen as an object window, the display status of which is to be changed;

a second step of sequentially selecting another window as a reference window among the rest of said windows in the display screen excepting said object window and comparing said object window with said another window selected as the reference window on the screen on the basis of the definition data stored in the third memory means to check whether said object window crosses said reference window;

a third step of dividing said object window into a subregion overlapped with said reference window and at least one rectangular non-overlapped subregion with the boundary of said overlapped subregion as a dividing line if said object window and said reference window cross each other;

a fourth step of comparing said non-overlapping subregion selected as an object rectangle with still another window selected sequentially as a reference window among the rest of said windows on the screen, and if said non-overlapped subregion selected as the object rectangle and said reference window cross each other, subdividing said non-overlapped subregion selected as object rectangle into a new subregion overlapped with said reference window and at least one rectangular non-overlapped subregion with the boundary of said new overlapped subregion as a dividing line, said comparing and subdivision being repeated by substituting said obtained non-overlapped subregion for a new object rectangular region until no more windows to be selected as said reference window remain; and

a fifth step of partially applying the display data of at least one partial area in said second memory means to a corresponding partial area in said first memory means, said partial area being divided in accordance with the resultant overlapped or non-overlapped subregions,

thereby changing the contents of display in the visible region occupied by the object window on the display screen.

7. A method of display control according to claim 1, wherein a window for which the display position on the display screen is to be shifted to the second position from the first position is designated as said object window in said first step, said second to fourth steps being executed for the object window at said second position, said method further comprising

a sixth step of determining a rectangular region in said object window at said first position corresponding to at least one non-overlapped subregion obtained for the object window at said second position, and

a seventh step of comparing the rectangular region obtained in said sixth step with another window selected as a reference window among the rest of the windows on the display screen, and when said rectangular region crosses said reference window,

dividing said rectangular region into a subregion overlapped with said reference window and at least one rectangular non-overlapped subregion with the boundary of said overlapped subregion as a dividing line, the comparing with other reference windows being repeated with the non-overlapped subregion as a new object rectangle,

said fifth step further comprising

an eighth step of shifting the display data of the final non-overlapped subregion obtained in said seventh step from the first position to the second position within said first memory means,

a ninth step of applying the contents of the virtual screen of said object window corresponding to the overlapped subregion obtained in said seventh step to the non-display region in the window at said second position, and

a tenth step of clearing the display data in the object window at the first position except for the overlapped subregion obtained in said seventh step.

19. A method according to claim 16, wherein when the specific one of the windows is to be deleted, all the windows other than said specific one window are designated as the reference windows in the third step and wherein in the seventh step, the non-overlapped area is cleared from any display data and display data for each of the overlapped subregions obtained in any of the fifth and sixth steps is selected from the basic data to be used for the window which is selectively used as the reference window in the step where that overlapped subregion is obtained.

20. A method according to claim 16, wherein when the specific one window is to be reduced in size by deleting a selected rectangular region of the window, said selected rectangular region of the specific one window is designated as the object window in place of the specific one window and all the windows other than said specific one window are designated as the reference windows in the third step and wherein in the seventh step, the non-overlapped area is cleared from any display data and display data for each of the overlapped subregions obtained in any of the fifth and sixth steps is selected from the basic data to be used for the window which is selectively used as the reference window in the step where that overlapped subregion is obtained.

☐ 6. Document ID: US 4660157 A

L11: Entry 6 of 8 File: USPT Apr 21, 1987

DOCUMENT-IDENTIFIER: US 4660157 A
TITLE: Real time video perspective digital map display method

Detailed Description Text (13):
The scene memory 35 is divided into scene subblocks, as seen in FIG. 3. Similar to the intermediate memory 20, the scene memory 35 has a size which is based on a terrain square with the aircraft located at a position which will provide sufficient terrain ahead of the aircraft for display purposes, and the size of the terrain square is a function of the range scale selected. Thus, the sides of the

h e b b g e e e f e f g ef b e

terrain square will be 6.4 km for a 8:1 and 51.2 km for a 1:1 scale. The elevation data section of the scene memory uses 8 bits per grid point to define the relative elevation. The absolute elevation of a grid point is related to the relative elevation stored in the scene memory by the following equation:

Detailed Description Text (66):
The clear circuit 180 responds to the output of scan generator 500 in the logic circuit 175 to clear the display memory 170 as data is read therefrom. Thus, when writing of data into any display unit of the display memory 170 begins, that memory unit will be completely clear, having been cleared during the previous read operation.

☐ 7. Document ID: US 4489389 A

L11: Entry 7 of 8 File: USPT Dec 18, 1984

DOCUMENT-IDENTIFIER: US 4489389 A
TITLE: Real time video perspective digital map display

Detailed Description Text (13):
The scene memory 35 is divided into scene subblocks, as seen in FIG. 3. Similar to the intermediate memory 20, the scene memory 35 has a size which is based on a terrain square with the aircraft located at a position which will provide sufficient terrain ahead of the aircraft for display purposes, and the size of the terrain square is a function of the range scale selected. Thus, the sides of the terrain square will be 6.4 km for a 8:1 and 51.2 km for a 1:1 scale. The elevation data section of the scene memory uses 8 bits per grid point to define the relative elevation. The absolute elevation of a grid point is related to the relative elevation stored in the scene memory by the following equation:

Detailed Description Text (68):
The clear circuit 180 responds to the output of scan generator 500 in the logic circuit 175 to clear the display memory 170 as data is read therefrom. Thus, when writing of data into any display unit of the display memory 170 begins, that memory unit will be completely clear, having been cleared during the previous read operation.

☐ 8. Document ID: US 4429387 A

L11: Entry 8 of 8 File: USPT Jan 31, 1984

DOCUMENT-IDENTIFIER: US 4429387 A
TITLE: Special character sequence detection circuit arrangement

Detailed Description Text (55):

h e b b g e e e f e f g ef b e

Each memory section furthermore, is divided into 16 memory subareas, each comprising eight memory locations which are represented by one half of a line of the pattern shown in FIG. 7. The memory subareas each are associated with a respective character number which is supplied to the address inputs A3 through A6 of the sequence detect PROM 304. Therefore, also the number of memory subareas is dependent upon the established number of special characters. The length of each memory subarea is determined by the specified length of the special character sequence. Here, each memory subarea consists of eight memory locations and any sequence can be specified by up to seven consecutively occurring special characters.

CLAIMS:

5. The special sequence detection circuit arrangement as recited in claim 4, wherein the read/write control unit comprises:

a first control flip-flop primed to be set by means of a first trigger pulse derived from a first timing pulse when occurring in combination with a character control signal being present when the presently selected line terminator is sending a data character, said control flip-flop having a clear input and an output carrying a latch control signal for enabling the memory buffer register, the second read only memory and subsequently the gating logic unit when a data character is received;

a first logical gate having inputs connected to receive the latch control signal and the first timing pulse and being designed for generating a write pulse enabling the random access memory to be written into upon the second occurrence of the first timing pulse as long as the latch control signal is present; and

a second control flip-flop having a preset input connected to receive the write pulse for being preset upon the occurrence of the write pulse, and designed for being reset thereafter upon occurrence of a second trigger pulse derived from a second timing pulse following the first timing pulse; wherein the clear input of the first control flip-flop is connected to receive said second trigger pulses for resetting the first control flip-flop.

h e b b g e e e f e f g ef b e